RÉPUBLIQUE DU BÉNIN
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ D'ABOMEY-CALAVI

INSTITUT DE FORMATION ET DE RECHERCHE EN INFORMATIQUE

BP 526 Cotonou Tel : +229 21 14 19 88
http://ifri-uac.bj Courriel : contact@ifri.uac.bj

MÉMOIRE

pour l'obtention du

Diplôme de Licence en Informatique

Option : Sécurité Informatique

Présenté par :
Jerry Erdhal **DADONOUGBO**

Mise en place d'une application de vote en ligne basée sur la blockchain

Sous la supervision :
Professeur Eugène C. **EZIN**, Enseignant-chercheur à l'Université d'Abomey-Calavi

Membres du jury :

Sylvain KPENAVOUN CHOGOU	Professeur	FSA	Président
Erick ADJE	Ingénieur	IFRI	Examinateur
Eugène C. EZIN	Professeur	IFRI et IMSP	Rapporteur

Année Académique : **2023-2024**

Sommaire

Dédicace ii

Remerciements iii

Résumé iv

Abstract v

Liste des figures vi

Liste des tableaux vii

Introduction 1

1 Etat de l'art 4

2 Méthodes et outils 13

3 Résultats et discussion 21

Conclusion 34

Bibliographie 39

Table des matières 40

Dédicace

À

Ma mère **Inès AÏMAWA**

Mon père **Serge DADONOUGBO**

Mes frères et sœurs **Allan, Rozenn, Kenzo et Kendall**

Remerciements

Je rends grâce à Dieu pour le souffle qu'Il m'a accordé jusqu'à ce jour. Je tiens à exprimer ma profonde reconnaissance à :

- Mon maître de mémoire, le professeur Eugène C. EZIN, directeur de l'IFRI et enseignant-chercheur à l'UAC, pour son encadrement, ses conseils avisés et son soutien tout au long de cette formation ;
- Le Directeur Adjoint de l'IFRI, Monsieur Gaston EDAH, pour son accompagnement et son soutien durant ce parcours académique ;
- Tous les professeurs de l'IFRI qui ont contribué à ma formation ;
- Tous les membres de l'administration de l'IFRI ;
- Toute l'équipe du bjCSIRT pour l'accueil chaleureux dont j'ai bénéficié ;
- Monsieur Gilbert AIMAWA, mon oncle, dont le soutien a été déterminant dans la concrétisation de ce projet. Sans lui, ce travail n'aurait jamais vu le jour ;
- Tous mes camarades et amis de l'IFRI ;
- Et enfin, vous qui lisez ces lignes.

Résumé

Dans ce projet, nous avons développé une application de vote électronique basée sur la technologie de la blockchain. Cette solution permet d'organiser des élections publiques et privées, garantissant la transparence, la sécurité, et la traçabilité des votes grâce aux "Smart Contracts". Elle intègre un système d'authentification via email et portefeuille MetaMask, permettant ainsi à chaque utilisateur de voter en toute sécurité. Le backend de l'application gère les appels à la blockchain via une API dédiée. Ce travail présente également les scénarios d'utilisation de l'application, incluant la création d'élections, le vote et l'affichage des résultats, tout en soulignant les potentielles limites liées aux coûts de déploiement.

Mots clés : Blockchain, vote électronique, Contrats Intelligents, transparence, sécurité, authentification, MetaMask, coûts de déploiement.

Abstract

In this project, we developed an electronic voting application based on blockchain technology. This solution allows the organization of public and private elections, ensuring transparency, security, and vote traceability through Smart Contracts. It includes an authentication system via email and MetaMask wallet, enabling each user to vote securely. The application's backend handles blockchain interactions through a dedicated API. This work also presents the usage scenarios of the application, including election creation, voting, and result display, while highlighting potential limitations related to deployment costs.

Key words: Blockchain, electronic voting, Smart Contracts, transparency, security, authentication, MetaMask, deployment costs.

Liste des figures

2.1 Architecture générale du système 14
2.2 Diagramme de cas d'utilisation du système 16
2.3 Diagramme de séquence du cas d'utilisation "Voter en Session Privée". 17

3.1 Résultat du déploiement du Smart Contrat 24
3.2 Résultat des tests unitaires 25
3.3 Formulaire d'inscription avec email, mot de passe et adresse Ethereum. 26
3.4 Formulaire de connexion classique par email et mot de passe. 27
3.5 Fenêtre MetaMask pour signer le message avec le nonce lors de la connexion. 27
3.6 Formulaire de création d'une élection avec champ pour le fichier CSV (si applicable). . 28
3.7 Confirmation de la création d'une élection. 29
3.8 Page de vote pour une élection privée 30
3.9 Confirmation du vote soumis pour l'élection privée. 30
3.10 Tableau récapitulatif des votes des candidats. 31

Liste des tableaux

1.1 Comparaison entre le vote traditionnel et le vote basé sur la blockchain 12

Introduction Générale

En Afrique de l'Ouest, le processus électoral est souvent marqué par une série de difficultés qui remettent en question la transparence et l'efficacité du vote. Dans des pays comme le Bénin, les élections sont ponctuées par des retards dans l'ouverture des bureaux de vote, une insuffisance de matériel électoral, et une mauvaise formation des membres du personnel électoral. Le **rapport de mission sur les élections générales nigérianes de février 2023**[4] souligne ces problématiques, notamment le début tardif des opérations de vote, les erreurs dans la distribution du matériel électoral, et une gestion logistique chaotique. Ces obstacles renforcent la méfiance des citoyens envers le processus électoral, affectant la participation et la crédibilité des résultats.

Au Bénin, les défis spécifiques sont tout aussi significatifs. Les **rapports des élections législatives de 2019**[1, 2] mettent en lumière :

- des retards récurrents dans la distribution des cartes d'électeurs et le déploiement du matériel électoral, entraînant des dysfonctionnements logistiques et des retards dans l'ouverture des bureaux de vote ;
- une formation insuffisante des agents électoraux, avec des disparités dans l'application des procédures le jour du scrutin ;
- un climat de méfiance alimenté par la politisation des institutions électorales et l'exclusion des partis d'opposition, compromettant l'inclusivité et la compétitivité des scrutins ;
- des incidents de violence électorale, exacerbés par des tensions sociales et régionales, qui compromettent la sécurité des agents et des électeurs ;
- une participation citoyenne en déclin, reflétant un désintérêt croissant et une méfiance envers le processus électoral.

Face à ces défis, la blockchain apparaît comme une solution technologique prometteuse. Grâce à son système décentralisé et immuable, la blockchain offre la possibilité de garantir une transparence totale tout au long du processus de vote. Chaque vote peut être enregistré de manière sécurisée et traçable, éliminant ainsi les risques de manipulation ou de fraude électorale. En outre, la blockchain permet de contourner les intermédiaires centralisés, réduisant ainsi la lourdeur administrative et les erreurs humaines.

1. Problématique

L'organisation des élections en Afrique de l'Ouest, particulièrement au Bénin, est confrontée à des défis majeurs qui compromettent l'intégrité du processus démocratique. Les dysfonctionnements observés lors des récentes élections, notamment les retards dans l'ouverture des bureaux de vote et les problèmes logistiques[1, 2], soulèvent des questions cruciales sur la fiabilité du système actuel. Face à ces enjeux, notre recherche s'articule autour de la question suivante :
Dans quelle mesure l'implémentation d'un système de vote électronique basé sur la blockchain peut-elle renforcer l'intégrité du processus électoral au Bénin, tout en prenant en compte les contraintes d'infrastructures technologiques et d'accessibilité ?
Cette problématique s'inscrit dans un contexte où les solutions technologiques innovantes, notamment la blockchain, émergent comme des alternatives prometteuses aux systèmes traditionnels de vote. Elle prend en considération non seulement les aspects techniques de la solution proposée, mais aussi les enjeux sociaux et infrastructurels spécifiques au contexte béninois.

Face à ces enjeux, notre recherche s'articule autour de la question suivante :
Dans quelle mesure l'implémentation d'un système de vote électronique basé sur la blockchain peut-elle renforcer l'intégrité du processus électoral au Bénin, tout en prenant en compte les contraintes d'infrastructures technologiques et d'accessibilité ?
Cette problématique s'inscrit dans un contexte où les solutions technologiques innovantes, notamment la blockchain, émergent comme des alternatives prometteuses aux systèmes traditionnels de vote. Elle prend en considération non seulement les aspects techniques de la solution proposée, mais aussi les enjeux sociaux et infrastructurels spécifiques au contexte béninois.

2. Contexte et justification

Les systèmes électoraux en Afrique de l'Ouest, et particulièrement au Bénin, souffrent d'un manque de transparence et de lourdeurs administratives. Les retards, les erreurs logistiques, et la mauvaise gestion des bureaux de vote sont autant de facteurs qui compromettent la confiance des électeurs et l'intégrité du processus. Parallèlement, le besoin d'une solution innovante et sécurisée pour améliorer le déroulement des élections est devenu crucial. La blockchain, en tant que technologie décentralisée et transparente, pourrait répondre à ce besoin en éliminant les problèmes de centralisation, en réduisant les erreurs humaines et en assurant la sécurité et la traçabilité des votes. L'adoption d'un tel système permettrait non seulement d'améliorer la confiance des citoyens, mais aussi de rendre le processus plus accessible, en particulier dans les zones où les infrastructures technologiques et administratives sont limitées. De plus, cette approche contribuerait à moderniser le système électoral, tout en répondant aux critiques concernant la lenteur et l'opacité des processus actuels.

3. Objectifs

Ce mémoire a pour objectif principal de proposer un modèle de système de vote électronique basé sur la blockchain pour le Bénin, en tenant compte des réalités locales et des défis spécifiques à la

région. Plus spécifiquement, il vise à :

- **analyser les faiblesses** du système électoral actuel, notamment en matière de transparence et de logistique ;
- **étudier les possibilités** d'adoption de la blockchain dans le processus électoral béninois ;
- **modéliser une solution** qui pourrait résoudre les problèmes de fraude, de lenteur administrative et d'erreurs humaines ;
- **proposer une architecture** technique adaptée aux besoins du pays ;
- **Évaluer la faisabilité** d'un tel système en termes d'infrastructures, d'inclusion et de coût.

4. Organisation du mémoire

Le présent mémoire fait le point sur nos travaux et est structuré en trois parties principales.

La première partie, intitulée *Contexte et état de l'art*, aborde les fondements théoriques des systèmes de vote électronique, la technologie blockchain, ainsi que les principales expériences de vote électronique dans d'autres pays. Cette section offre une vue d'ensemble sur les bases du vote électronique et l'intérêt de la blockchain dans ce domaine.

La deuxième partie, *Méthode et outils*, propose une analyse approfondie des défis du système électoral au Bénin, en mettant en lumière les faiblesses logistiques, administratives, et les problématiques de transparence et de sécurité. Ensuite, cette partie détaille la proposition d'un modèle de vote électronique basé sur la blockchain, avec une description de l'architecture technique et des technologies utilisées, telles que les smart contracts.

La troisième partie, *Résultats et discussion*, présente une étude de faisabilité, en analysant les enjeux d'infrastructure, d'inclusion technologique, ainsi que les défis liés à l'adoption d'un tel système au Bénin. Cette partie conclut par une synthèse des résultats et des recommandations pour une mise en œuvre efficace du système proposé.

Chapitre 1

Etat de l'art

Introduction

Ce premier chapitre est consacré à la revue de la littérature, essentielle pour comprendre les fondements théoriques et les pratiques actuelles en matière de systèmes de vote et de technologies associées. Dans un contexte où les processus électoraux en Afrique de l'Ouest, et notamment au Bénin, sont souvent confrontés à des défis importants tels que les retards, les insuffisances matérielles et les fraudes, il est crucial de situer notre projet dans un cadre analytique solide. Nous commencerons par une analyse des systèmes de vote traditionnels en Afrique de l'Ouest, mettant en lumière les problématiques administratives et logistiques rencontrées lors des élections. Ensuite, nous introduirons la blockchain, en expliquant ses principes de base et ses applications au-delà des cryptomonnaies, afin de préparer le terrain pour sa potentialité dans le domaine électoral. Enfin, nous examinerons les initiatives actuelles utilisant la blockchain pour le vote, en discutant des avantages et des défis de cette technologie.

Cette revue de la littérature vise à établir les bases nécessaires pour évaluer comment la blockchain pourrait offrir des solutions innovantes aux défis des systèmes de vote traditionnels et améliorer la transparence et la fiabilité des élections dans des contextes similaires à celui du Bénin.

1.1 Les systèmes de vote traditionnels

1.1.1 Le processus électoral classique en Afrique de l'Ouest

Le Bénin, tout comme de nombreux pays de la région ouest-africaine, a vu ses systèmes de vote traditionnels évoluer au fil du temps, reflet des processus démocratiques en constante transformation. Ce sous-chapitre se concentre sur les étapes principales du processus électoral au Bénin[6], qui incarne l'engagement du pays envers la démocratie et la participation active de ses citoyens à la gouvernance.

- **Inscription des électeurs** : Le processus électoral commence par l'inscription des électeurs. Les citoyens éligibles doivent se faire enregistrer afin de figurer sur les listes électorales. Cette étape implique la fourniture de documents d'identité ainsi que des informations personnelles pour

garantir que seuls les citoyens remplissant les conditions requises puissent participer au vote. L'objectif de cette phase est d'assurer une participation inclusive et équitable de la population à l'élection.

- **Période de campagne** : Avant le jour du scrutin, une période de campagne électorale s'ouvre, où les partis politiques et les candidats mènent des actions de sensibilisation et de persuasion auprès des électeurs. Les activités de campagne incluent des meetings, des discours publics, des débats et d'autres événements visant à promouvoir les programmes électoraux des candidats. C'est une période décisive pour que les électeurs puissent s'informer et évaluer les propositions des différents partis.

- **Jour de l'élection** : Le jour du scrutin, les électeurs inscrits se rendent aux bureaux de vote pour exprimer leur choix en toute confidentialité. Le processus de vote est supervisé par des agents électoraux qui veillent à ce que les opérations se déroulent dans l'ordre et le respect des règles établies. Le jour de l'élection marque le point culminant de la campagne électorale et offre aux citoyens l'opportunité d'exercer leur droit démocratique.

- **Comptage des voix** : Après la clôture des bureaux de vote, les bulletins sont collectés et comptés de manière transparente par les autorités électorales, avec la présence éventuelle de représentants des partis politiques pour assurer l'équité du processus. Le décompte minutieux des voix est essentiel pour garantir l'exactitude des résultats et la légitimité du processus.

- **Annonce des résultats** : Une fois les voix comptabilisées, les résultats sont annoncés publiquement. Cette étape est cruciale pour la transparence du processus et l'acceptation des résultats par toutes les parties. L'annonce officielle des résultats détermine les nouveaux élus et reflète la volonté populaire exprimée lors des élections.

Le système électoral béninois repose sur un cadre multipartite, et le pays organise à la fois des élections présidentielles et législatives. La législature est composée de l'Assemblée nationale avec 83 députés élus directement, tandis que l'élection présidentielle suit un système à deux tours si nécessaire. À travers ces différentes étapes, le processus électoral au Bénin illustre l'importance des pratiques démocratiques et le rôle central des citoyens dans le façonnement du paysage politique.

1.1.2 Les défis du processus électoral en Afrique de l'Ouest et au Bénin

En Afrique de l'Ouest, et plus particulièrement au Bénin, le processus électoral rencontre de nombreux défis qui affectent la transparence, la rapidité et l'efficacité des élections. Ces difficultés, bien documentées dans divers rapports[1, 2], sont souvent liées à des problèmes logistiques, administratifs et organisationnels qui ralentissent le déroulement des scrutins et limitent la participation des électeurs. Voici les principaux obstacles identifiés :

- **La lourdeur administrative et logistique** : - Les retards dans l'ouverture des bureaux de vote, souvent causés par l'arrivée tardive du matériel électoral ou des membres des bureaux, sont une récurrence lors des scrutins. Ces dysfonctionnements ont été observés au Bénin lors des législatives de 2019, où le déploiement du matériel a été tardif dans plusieurs localités[1]. - Par ailleurs, des erreurs dans la répartition du matériel et une mauvaise coordination entre les acteurs électoraux aggravent ces retards, ce qui nuit à la crédibilité du processus.

- **Le manque de formation et de compétences du personnel électoral** : - La formation des agents électoraux est souvent inadéquate, car réalisée de manière précipitée ou selon une méthode en cascade. Les agents de terrain ne maîtrisent pas toujours les procédures ni les équipements mis à leur disposition[1]. - Ce manque de préparation génère des incohérences dans l'application des règles électorales et des lenteurs dans les opérations de vote, affectant directement la confiance des électeurs dans le système.

- **L'accessibilité et l'inclusion** : - Les zones rurales et reculées du Bénin souffrent d'un manque d'infrastructures essentielles comme l'électricité, les moyens de transport, et l'accès à Internet. Ces limitations empêchent une partie de la population d'accéder facilement aux bureaux de vote, marginalisant des communautés entières[2]. - Les personnes à mobilité réduite, les jeunes et les femmes sont également souvent sous-représentés dans les processus électoraux[2].

- **Les risques de fraude et de manipulation** : - La fraude électorale demeure une préoccupation majeure. Elle se manifeste par des pratiques telles que l'altération des listes électorales, la destruction de bulletins de vote ou l'intimidation des électeurs[1]. - Au Bénin, le climat de méfiance est renforcé par une politisation excessive des institutions électorales et des tensions sociales autour des scrutins, comme cela a été constaté en 2019[2].

- **La lenteur du comptage des voix et l'annonce des résultats** : - Le dépouillement des votes et la centralisation des résultats sont souvent longs et chaotiques. Lors des législatives de 2019, le manque de dispositifs de sécurisation des résultats a suscité des suspicions de manipulation[2]. - Cette lenteur alimente parfois des tensions politiques, des accusations de fraude et des violences post-électorales, compromettant la stabilité du pays.

Ces défis structurels et organisationnels mettent en lumière l'urgence d'une réforme profonde des systèmes électoraux dans la région. Au Bénin, en particulier, l'introduction de technologies comme la blockchain pourrait offrir des solutions novatrices pour surmonter ces obstacles, en garantissant un processus plus transparent, sécurisé et inclusif.

1.2 La blockchain : une technologie émergente

1.2.1 Définition de la blockchain et principes de base

La blockchain, littéralement « chaîne de blocs », est une technologie de stockage et de transmission d'informations, transparente, sécurisée et fonctionnant sans organe central de contrôle. Elle se définit comme un registre distribué et immuable, constitué de blocs de données liés entre eux de manière chronologique et sécurisée par des algorithmes cryptographiques.[7] Ce registre est partagé entre plusieurs utilisateurs, appelés « nœuds », et chaque modification apportée aux données est automatiquement visible pour l'ensemble des participants du réseau.
L'un des principes fondamentaux de la blockchain est qu'une fois qu'un bloc est validé et ajouté à la chaîne, il devient pratiquement impossible de le modifier ou de le supprimer. Cela garantit l'intégrité et la permanence des données, ce qui en fait un outil fiable pour des applications nécessitant des enregistrements infalsifiables, comme le vote électronique.

- **Fonctionnement décentralisé, immuabilité et transparence** : Contrairement aux systèmes traditionnels centralisés, où une autorité unique contrôle les informations et les transactions, la

blockchain repose sur un modèle décentralisé. Chaque nœud du réseau possède une copie du registre, et les décisions de validation des transactions ou des blocs se font par consensus entre les participants. Ce mécanisme de consensus assure que les données inscrites dans la blockchain sont validées de manière collective, réduisant ainsi le risque de fraudes ou de manipulations.
L'immuabilité est un autre aspect clé de la blockchain. Une fois qu'une transaction ou une information est enregistrée dans un bloc validé, elle ne peut plus être altérée. Cela apporte une sécurité supplémentaire aux systèmes qui nécessitent des garanties contre la falsification. Enfin, la transparence est assurée par le fait que toutes les transactions sont accessibles aux membres du réseau. Bien que les données puissent être chiffrées pour des raisons de confidentialité, leur traçabilité et leur enregistrement restent visibles et accessibles à tous.

- **Les différentes générations de blockchain** : L'évolution de la technologie blockchain peut être divisée en trois générations, chacune apportant des innovations majeures et élargissant les possibilités offertes par cette technologie révolutionnaire. [5]

Première génération : Bitcoin

La première génération de blockchain est représentée par le réseau Bitcoin, créé en 2009 par Satoshi Nakamoto. Le Bitcoin visait à offrir une alternative décentralisée au système monétaire traditionnel, permettant des transactions de pair à pair sans l'intervention d'une autorité centrale, comme une banque. L'innovation clé du Bitcoin réside dans son protocole de preuve de travail (Proof of Work), qui permet de sécuriser le réseau et de valider les transactions sans nécessiter de confiance entre les participants. La blockchain de Bitcoin a rendu les transactions financières plus accessibles, anonymes, et sécurisées, tout en réduisant la dépendance aux institutions financières traditionnelles.
Cependant, Bitcoin se limite à la transmission de valeur et n'offre pas la possibilité de programmer des conditions sur les transactions. C'est ce besoin qui a conduit à la naissance de la blockchain de deuxième génération.

Deuxième génération : Ethereum

En 2015, **Ethereum** a été introduit comme une blockchain de deuxième génération, offrant bien plus que de simples transactions financières. Ethereum a inauguré une **plateforme décentralisée** permettant aux développeurs de créer des **contrats intelligents** (smart contracts), des **applications décentralisées** (dApps), et de gérer des **tokens**. Les **contrats intelligents** sont des accords automatisés qui s'exécutent lorsque certaines conditions prédéfinies sont remplies, sans avoir besoin d'un tiers. Cette innovation a ouvert la voie à de nouveaux cas d'usage pour la blockchain, tels que la **finance décentralisée** (DeFi), les **NFTs**, et les **stablecoins**.
Ethereum a aussi introduit un modèle de gouvernance où les utilisateurs participent à la gestion du réseau via des droits de vote intégrés dans le protocole.

Troisième génération : Scalabilité et interopérabilité

Les blockchains de troisième génération, comme **Cardano**, **Polkadot**, et **Ethereum 2.0**, visent à résoudre deux des principaux défis auxquels sont confrontées les blockchains : la scalabilité et l'interopérabilité.

- **Scalabilité** : Alors que le nombre d'utilisateurs de blockchain croît rapidement, la capacité de traitement des transactions devient un enjeu majeur. Les blockchains de premières et deuxièmes générations, comme Bitcoin et Ethereum, sont limitées en termes de transactions par seconde. Cela entraîne des frais élevés et une lenteur du réseau. Les nouvelles blockchains utilisent des mécanismes comme la **preuve d'enjeu (Proof of Stake)** pour améliorer l'efficacité énergétique et augmenter le nombre de transactions traitées simultanément.
- **Interopérabilité** : Une autre innovation clé de cette génération est la capacité pour différentes blockchains de communiquer entre elles. Cette **interopérabilité** est essentielle pour créer un écosystème blockchain fluide, où des actifs numériques peuvent être transférés d'une chaîne à l'autre sans limitations.

1.2.2 Les applications de la blockchain au-delà des cryptomonnaies

La technologie blockchain, bien que popularisée par les cryptomonnaies, offre une multitude d'applications dans divers secteurs au-delà du domaine financier.[8] Grâce à ses caractéristiques uniques, telles que la décentralisation, l'immuabilité, et la transparence, la blockchain est utilisée dans plusieurs industries pour résoudre des problèmes complexes de gestion des données, de traçabilité et de sécurité.

a) **La blockchain dans le secteur bancaire**
Dans le secteur bancaire, la blockchain joue un rôle essentiel dans la certification des opérations effectuées par les clients. Grâce à la dématérialisation des services bancaires, les transactions peuvent être effectuées via un téléphone portable, facilitant ainsi les opérations à faible coût dès qu'une connexion internet est disponible. L'objectif principal de l'intégration de la blockchain dans le secteur bancaire est de **faciliter le transfert d'argent** de manière plus rapide et moins coûteuse.

b) **La blockchain dans le secteur juridique**
Dans le secteur juridique, l'enjeu réside dans la **dématérialisation** des documents légaux tout en assurant une protection accrue des données sensibles. Les plateformes basées sur la blockchain permettent de regrouper tous les documents juridiques nécessaires de manière sécurisée, garantissant la confidentialité et l'intégrité des informations.
La société Axiocap propose des solutions de gestion dématérialisée des documents et des titres des entreprises, permettant de créer des **registres sécurisés** via la blockchain.

c) **La blockchain dans le secteur de l'assurance**
Avec les **smart contracts**, la blockchain simplifie les procédures d'indemnisation dans le secteur de l'assurance. Cela permet un traitement rapide des remboursements sans intervention humaine. Par exemple, **AXA** a mis en place l'application **Fizzy**, qui utilise la blockchain pour indemniser automatiquement les passagers en cas de retard d'avion. L'application suit les horaires en temps réel et exécute automatiquement le paiement en cas de retard, sans nécessiter de justificatifs supplémentaires de la part du client.

d) **La blockchain dans le secteur de la santé**
La blockchain permet de stocker et de partager de manière sécurisée les **données médicales** des patients, accessibles via un téléphone portable. Cela facilite la gestion des informations pour les patients, tout en garantissant la confidentialité des données sensibles pour le personnel de santé. Des entreprises comme **Patientory** ou **Dokchain** offrent déjà des solutions pour permettre un accès facile et sécurisé aux dossiers médicaux.

e) **La blockchain dans le secteur alimentaire**
La blockchain dans l'industrie alimentaire permet de garantir la **traçabilité des produits**, en fournissant aux consommateurs des informations transparentes sur la provenance, la transformation, et l'empaquetage des aliments. En scannant un QR code sur un produit, les consommateurs peuvent accéder à toutes les informations relatives au parcours de l'aliment, assurant ainsi la véracité des informations et l'absence de falsification.

f) **La blockchain dans le secteur étatique**
La blockchain est également utilisée par les gouvernements pour **gérer de manière sécurisée** les données sensibles des citoyens. Un des exemples les plus avancés est celui de l'Estonie, qui utilise la blockchain pour centraliser et standardiser les données personnelles des citoyens, dans des domaines tels que la santé et la justice. Cette technologie pourrait également être appliquée dans le cadre des **élections en ligne**, offrant des processus de vote transparents, sécurisés, et facilement vérifiables.

1.3 Blockchain et systèmes électoraux

1.3.1 Cas d'études : Les initiatives de vote électronique basé sur la blockchain

Plusieurs pays ont expérimenté l'utilisation de la blockchain pour leurs systèmes électoraux, principalement dans un souci de transparence, de sécurité, et de vérifiabilité des votes. Voici quelques exemples notables :

Exemples de pays ayant utilisé un système basé sur la blockchain pour des élections

Plusieurs pays ont expérimenté l'utilisation de la blockchain pour leurs systèmes électoraux, principalement dans un souci de transparence, de sécurité, et de vérifiabilité des votes. Voici quelques exemples notables :

- **Sierra Leone**
  En 2018, la Sierra Leone a fait la une des journaux en devenant le premier pays au monde à tester la blockchain pour une élection présidentielle. Le projet pilote a été mené par l'organisation **Agora**[9] qui a utilisé la blockchain pour surveiller et valider les résultats électoraux dans certaines zones. Bien que le vote lui-même n'ait pas eu lieu sur la blockchain, cette technologie a été utilisée pour garantir l'intégrité et la transparence du dépouillement.

- **Russie**
  En 2020, la Russie a testé la blockchain pour ses élections locales et régionales. Le système, développé par la société **Waves Enterprise**[10], a permis à certains électeurs de Moscou et de

la région de Nizhny Novgorod de voter en ligne. L'objectif était d'assurer que chaque vote soit comptabilisé de manière transparente et que le processus soit auditables sans compromettre la confidentialité des électeurs.

- **États-Unis**
  Aux États-Unis, plusieurs États, tels que la Virginie-Occidentale, ont testé l'utilisation de la blockchain pour le vote des citoyens vivant à l'étranger ou des militaires. En 2018, l'application mobile **Voatz**[11] a permis à des électeurs de voter en utilisant un système sécurisé basé sur la blockchain. Bien que cette initiative ait rencontré des critiques concernant la sécurité, elle a ouvert le débat sur la viabilité de la blockchain dans les systèmes électoraux américains.

1.3.2 Analyse des résultats et des défis rencontrés dans ces expériences

Résultats observés

L'utilisation de la blockchain pour le vote électronique a permis de révéler plusieurs avantages concrets dans diverses expériences internationales. Ces résultats montrent que cette technologie peut transformer le processus électoral de manière significative :

- **Sécurité et intégrité des votes :** L'expérience de la Sierra Leone lors des élections présidentielles de 2018 a démontré l'efficacité de la blockchain dans le renforcement de la sécurité électorale. Le système mis en place par Agora a permis de valider et d'enregistrer les résultats de manière immuable dans la région ouest du pays, offrant ainsi une traçabilité complète du processus de dépouillement. Cette expérience a notamment montré une réduction significative des contestations liées aux résultats dans les zones où la blockchain était utilisée, grâce à la transparence et à la vérifiabilité inhérentes à cette technologie.

- **Transparence accrue :** Grâce à la blockchain, les transactions électorales sont traçables et auditées par les autorités compétentes. En Sierra Leone, l'organisation Agora a utilisé la blockchain en 2018 pour surveiller et valider les résultats électoraux dans certaines régions. Cette transparence accrue renforce la confiance des citoyens dans le processus électoral et aide à prévenir la fraude électorale.

- **Amélioration de la vérifiabilité :** La traçabilité inhérente à la blockchain permet de résoudre un problème clé du vote électronique traditionnel : la vérifiabilité. Les citoyens et les autorités peuvent retracer l'intégralité du processus de vote sans compromettre la sécurité du système, comme cela a été démontré dans plusieurs expériences, notamment en Estonie.

- **Applications élargies :** Outre le vote lui-même, la blockchain a montré qu'elle pouvait également jouer un rôle clé dans les étapes de comptage et de validation des résultats électoraux. L'expérience de la Sierra Leone a mis en lumière le potentiel de la blockchain dans le dépouillement des votes, permettant un comptage transparent et fiable des bulletins.

Défis rencontrés

Malgré les nombreux avantages, l'adoption de la blockchain dans le cadre des élections fait face à plusieurs défis majeurs, tant sur le plan technique que social :

- **Équilibre entre transparence et confidentialité :** L'expérience russe en 2020 a révélé un défi crucial : comment concilier la transparence, essentielle pour garantir un processus électoral juste et auditable, avec le respect de la confidentialité des électeurs, un principe fondamental de la démocratie. Le système de Waves Enterprise a soulevé des préoccupations quant à la protection de l'anonymat des votants.
- **Fiabilité des solutions techniques :** Aux États-Unis, l'utilisation de l'application Voatz a suscité des doutes sur la sécurité des systèmes de vote électronique basés sur la blockchain. Malgré ses promesses de sécurité, cette application a montré que la blockchain, bien que robuste, n'est pas exempte de vulnérabilités. Des tests approfondis sont nécessaires pour s'assurer que ces systèmes respectent les standards de sécurité les plus stricts.
- **Infrastructures et inclusion numérique :** Dans des pays comme la Sierra Leone, l'accès limité à Internet dans certaines régions a freiné l'utilisation de la blockchain pour les élections. En Russie et aux États-Unis, des questions d'accessibilité ont également été soulevées, en particulier pour les populations vulnérables ou celles n'ayant pas accès à une connexion Internet fiable. Ces obstacles risquent d'exacerber les inégalités en matière de participation électorale.
- **Résistance au changement :** Dans plusieurs pays, la méfiance des citoyens et des autorités électorales envers la blockchain constitue un obstacle majeur à son adoption. La crainte de cyberattaques et l'incertitude liée à cette technologie nouvelle ont ralenti son déploiement à grande échelle, en Russie et aux États-Unis notamment. Des efforts substantiels en matière de sensibilisation et d'éducation sont nécessaires pour surmonter cette réticence.

Pour mieux comprendre les avantages et les limites du vote électronique basé sur la blockchain, il est pertinent de le comparer au vote traditionnel. Le tableau suivant met en évidence les principales différences entre ces deux systèmes, en tenant compte des aspects techniques, sécuritaires et organisationnels.

TABLE 1.1 : Comparaison entre le vote traditionnel et le vote basé sur la blockchain

Critères	Vote traditionnel	Vote basé sur la blockchain
Sécurité	Dépend de la surveillance humaine et des systèmes locaux. Vulnérable aux fraudes, erreurs humaines, et manipulations.	Sécurisé par des mécanismes cryptographiques avancés et des registres immuables. Réduit le risque de falsification et de perte de données.
Transparence	Limitée. Les citoyens ont souvent peu de visibilité sur le processus de dépouillement et de validation des résultats.	Transparence accrue grâce à des registres accessibles et audités en temps réel par toutes les parties prenantes.
Traçabilité	Peu ou pas de traçabilité. Une fois le vote déposé, son suivi devient impossible ou opaque.	Traçabilité complète. Chaque vote est enregistré sur la blockchain et peut être vérifié tout au long du processus électoral.
Coût	Nécessite des ressources physiques (bulletins, urnes, personnel, transport), ce qui peut entraîner des coûts élevés.	Coûts initiaux élevés pour la mise en place, mais réduction des coûts opérationnels à long terme (moins de papier, moins de personnel).
Accessibilité	Peut être difficile pour les personnes vivant à l'étranger ou dans des zones reculées.	Accessible à distance via des dispositifs connectés, augmentant ainsi la participation électorale.

Conclusion

Ce chapitre a exploré les systèmes de vote traditionnels en Afrique de l'Ouest, en particulier au Bénin, et les défis qu'ils posent, notamment les problèmes logistiques, les fraudes électorales et la perte de confiance des électeurs. Cette analyse met en évidence la nécessité d'innovations, préparant ainsi le terrain pour l'introduction de solutions technologiques avancées telles que la blockchain. La blockchain, en tant que technologie décentralisée et transparente, a été examinée comme une solution potentielle pour surmonter ces obstacles. Ses applications dans divers secteurs et ses avantages dans les systèmes électoraux ont été discutés, tout en reconnaissant les défis, tels que l'inclusion numérique et les infrastructures nécessaires.
Cette analyse prépare le terrain pour le prochain chapitre, qui abordera en détail la solution innovante proposée ainsi que la méthodologie et les choix technologiques pour un système de vote électronique basé sur la blockchain.
Cette solution innove en rendant l'usage de la blockchain transparent pour l'utilisateur final : Contrairement aux systèmes classiques de vote blockchain qui exigent des votants qu'ils possèdent un portefeuille et une adresse blockchain, ici, cette contrainte est supprimée.

Chapitre 2

Méthodes et outils

Introduction

Ce chapitre se concentre sur la modélisation et le développement du système de vote basé sur la blockchain. Nous commencerons par la modélisation, en utilisant des diagrammes pour décrire la structure et les relations des différentes entités du système. Nous passerons ensuite au développement, en détaillant la conception des smart contracts, le backend et le frontend. Enfin, nous aborderons les tests et la validation pour assurer le bon fonctionnement du système. Ce chapitre fournit une vue d'ensemble des étapes essentielles pour créer un système de vote sécurisé et efficace.

2.1 Architecture générale du système

L'architecture du système de vote électronique repose sur trois composantes principales : une interface utilisateur pour les votants, une interface administrateur pour les autorités et autres administrateurs (particuliers), et des contrats intelligents déployés sur une blockchain publique. Le backend fait le lien entre ces interfaces et la blockchain, garantissant que les votes sont stockés et comptabilisés de manière sécurisée et transparente. Le schéma ci-dessous illustre cette architecture.

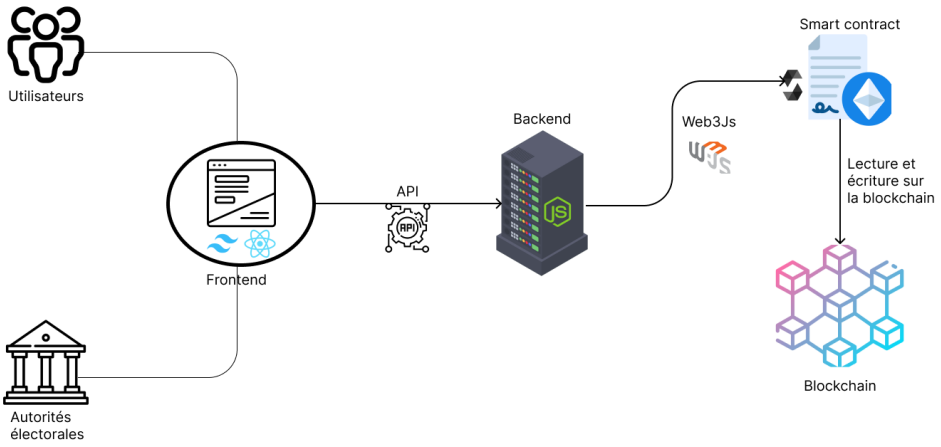


FIGURE 2.1 : Architecture générale du système

Acteurs principaux du système

Dans le cadre de notre système de vote électronique basé sur la blockchain, plusieurs types d'utilisateurs interagissent avec la plateforme, chacun ayant des rôles et responsabilités distincts. Ces acteurs sont essentiels pour garantir le bon fonctionnement et la transparence des élections. Les trois principaux groupes d'utilisateurs sont :

a) **Votants** : Ce groupe constitue les citoyens qui participent au processus électoral. Leur rôle principal est de voter lors des sessions de vote, qu'elles soient privées ou publiques. Ils doivent suivre les procédures d'authentification ou de fourniture d'informations selon le type de session pour exercer leur droit de vote.

b) **Autorités Électorales** : Les autorités électorales sont responsables de la création, de la gestion et de la supervision des élections. Elles configurent les sessions de vote, définissent les paramètres, et assurent la conformité aux règles établies. Elles gèrent également les listes électorales pour les sessions privées et supervisent le bon déroulement des élections.

c) **Particuliers** : Ce groupe, constitué de particuliers ayant créé un compte ou s'étant authentifiés, est chargé de créer et de gérer des sessions de vote publiques. Bien qu'ils ne disposent pas des privilèges d'administration du système, ils jouent un rôle crucial dans la mise en place des sessions ouvertes au grand public, ajoutant des candidats et définissant les modalités de vote.

Chaque acteur joue un rôle spécifique dans le processus électoral, contribuant ainsi à la réussite et à l'intégrité du système de vote électronique.

Composants Logiciels

Le système comporte plusieurs composantes, chacune jouant un rôle essentiel dans le bon fonctionnement de l'application. Les principales composantes sont les suivantes :

- **Front-end (Interface utilisateur)** : Cette partie du système permet aux utilisateurs d'interagir facilement avec la plateforme. Elle offre des interfaces distinctes pour :

  - **Les autorités électorales** qui peuvent créer et gérer des élections privées. Ces autorités disposent d'un espace pour importer une liste électorale, ajouter des candidats avec leurs photos et descriptions, et gérer le processus électoral.
  - **Les particuliers** qui peuvent aussi créer des élections publiques sans nécessité de liste électorale. Ils peuvent renseigner leurs informations pour participer à une session de vote ouverte à tous.
  - **Les votants** qui accèdent à une interface dédiée pour s'authentifier et voter. Selon le type de session, ils peuvent soit s'authentifier via une liste électorale pour les sessions privées, soit renseigner des informations personnelles pour les sessions ouvertes.

- **Back-end (Serveur)** : Le back-end assure la gestion des données et des transactions. Il communique avec la blockchain pour le stockage et la comptabilisation des votes, tout en restant indépendant pour les processus d'authentification. Ses composantes principales sont :
  - **Base de données traditionnelle** : Stocke les informations liées aux utilisateurs (autorités électorale et créateurs d'élections).
  - **Gestion des sessions électorales** : Le serveur contrôle l'accès aux sessions électorales en fonction de l'authentification des utilisateurs. Une fois un vote effectué, il est envoyé à la blockchain pour être comptabilisé.

- **Blockchain (Ethereum)** : La blockchain Ethereum a été choisie pour son **écosystème bien établi**, sa **maturité technologique** et la **large disponibilité d'outils et de bibliothèques de développement**. Elle est utilisée pour stocker et comptabiliser les votes, chaque vote étant enregistré sous forme de transaction, assurant ainsi son immutabilité et sa traçabilité. Cette approche permet de bénéficier de la sécurité cryptographique et de la transparence inhérentes à la blockchain Ethereum. Les garanties offertes incluent :
  - Les votes ne peuvent pas être modifiés ou supprimés une fois comptabilisés, assurant l'intégrité des données.
  - Le résultat final est transparent et accessible à toutes les parties concernées, renforçant la confiance des électeurs et des parties prenantes.

- **Smart Contracts** : Les smart contracts gèrent l'exécution automatique des élections. Ils encadrent plusieurs aspects :
  - **Création d'une élection** : Enregistrement des candidats, des sessions de vote (privées ou ouvertes), et des conditions spécifiques de chaque élection.
  - **Vérification des votes uniques** : Pour les élections privées, les smart contracts s'assurent qu'un votant inscrit sur la liste ne vote qu'une seule fois.
  - **Comptabilisation automatique** : À la fin du scrutin, le smart contract effectue le comptage des votes et publie les résultats de manière automatisée.

- **Authentification et gestion des listes électorales** : Pour les sessions privées, les autorités électorales importent une liste d'électeurs, et les utilisateurs s'authentifient avant de voter. Ce processus garantit que chaque personne sur la liste ne vote qu'une seule fois. Pour les sessions ouvertes à tous, les utilisateurs fournissent des informations personnelles minimales et peuvent voter sans besoin d'authentification stricte.

2.2 Modélisation du système

La modélisation est la première étape vers la construction d'un système de vote électronique basé sur la blockchain. Elle permet de structurer les composants et les interactions du système avant le développement. En fournissant une vue d'ensemble claire, elle guide le processus de conception et garantit que chaque élément est bien aligné avec les exigences fonctionnelles et techniques.

2.2.1 Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation illustre les interactions entre les différents acteurs du système et les principales fonctionnalités à implémenter. Il représente de manière simple et visuelle comment les utilisateurs interagissent avec le système de vote.
La figure 2.2 illustre le diagramme de cas d'utilisation de notre système :

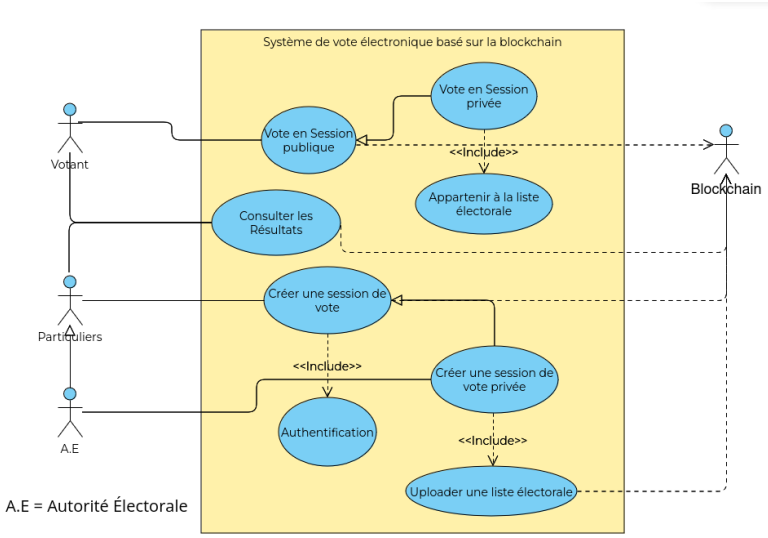


FIGURE 2.2 : Diagramme de cas d'utilisation du système

Les **Votants** peuvent participer à des sessions de vote publiques ou privées. Dans le cas des sessions privées, ils doivent figurer sur une liste électorale validée par les **Autorités Électorales**. Ces dernières sont responsables de la gestion des sessions de vote privées, notamment en créant ces sessions et en téléchargeant les listes électorales correspondantes. Elles jouent également un rôle dans la vérification des résultats et dans l'organisation globale des élections.
Les **Particuliers** (ou créateurs de sessions de vote) peuvent, après authentification, créer des sessions de vote publiques. Ils ne sont pas de véritables administrateurs mais disposent de l'autorisation de créer et gérer des sessions accessibles à tous les votants sans la contrainte d'une liste électorale.
Chaque acteur dispose d'un ensemble de fonctionnalités spécifiques, illustrant ainsi la répartition des responsabilités et les conditions d'accès aux différents types de sessions de vote.

2.2.2 Diagramme de séquence

Le diagramme de séquence permet de modéliser le déroulement des interactions entre les acteurs et les différents composants du système dans le temps. Il décrit le flux des messages échangés entre les utilisateurs (votants, autorités électorales, créateurs de sessions) et les composants du système de

vote basé sur la blockchain, depuis la création d'une session de vote jusqu'à la validation des résultats. La figure 2.3 illustre le diagramme de séquence du cas d'utilisation **"Voter en Session Privée"** du système proposé.

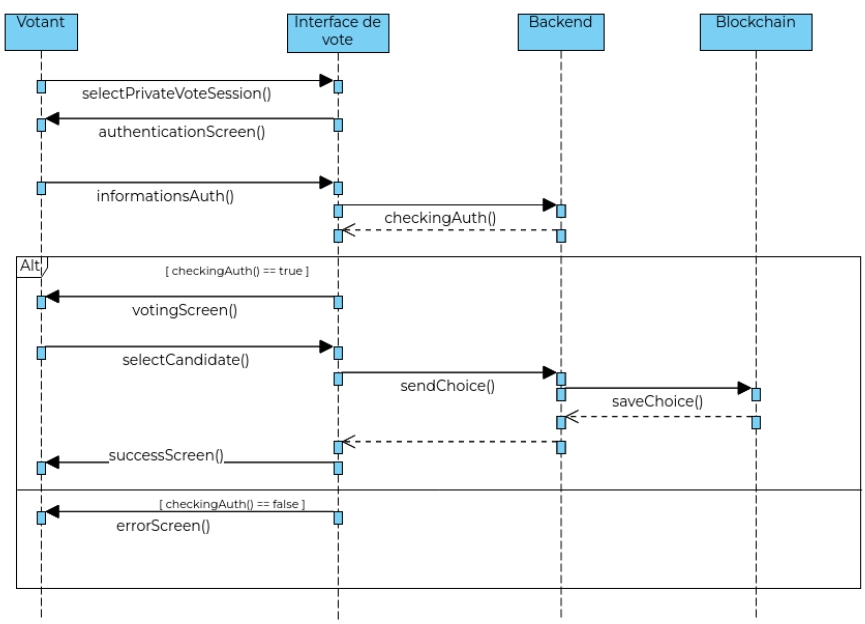


FIGURE 2.3 : Diagramme de séquence du cas d'utilisation "Voter en Session Privée".

Ce diagramme de séquence illustre le processus d'interaction entre un votant et les différentes composantes du système de vote électronique basé sur la blockchain. Il décrit l'ensemble des étapes nécessaires à la soumission et à l'enregistrement d'un vote. Les éléments suivants participent à cette interaction : le **votant**, l'**interface utilisateur**, le **serveur backend** qui gère la logique du système, et la **blockchain** qui enregistre de manière immuable les transactions de vote.

Acteurs Principaux

- **Votant** : L'utilisateur authentifié qui souhaite participer à une session de vote.
- **Interface de Vote** : Application front-end où le votant interagit avec le système.
- **Serveur Backend** : Le système backend qui gère les requêtes du votant et les interactions avec la blockchain.
- **Blockchain** : Le registre immuable où les votes sont enregistrés sous forme de transactions.

Description du processus

- **Sélection de la session de vote privée** : Le votant commence par sélectionner une session de vote privée à partir de l'interface de vote, ce qui déclenche l'affichage de l'écran d'authentification. Cette étape est essentielle pour restreindre l'accès aux sessions de vote privées aux utilisateurs enregistrés sur une liste électorale préalablement définie par les autorités électorales.

- **Authentification** : Une fois que l'utilisateur entre ses informations d'authentification, celles-ci sont envoyées à l'interface de vote, qui les transmet ensuite au back-end pour vérification via la fonction `checkingAuth()`.
  - Si l'authentification est réussie (`checkingAuth() == true`), l'interface affiche l'écran de vote (`votingScreen()`) pour permettre au votant de sélectionner un candidat.
  - Si l'authentification échoue (`checkingAuth() == false`), un écran d'erreur (`errorScreen()`) est affiché pour informer le votant de l'échec de l'opération.
- **Sélection du candidat** : Une fois authentifié avec succès, le votant accède à l'écran de vote, où il peut sélectionner son candidat préféré. Cette action est capturée par l'interface de vote via la fonction `selectCandidate()`. Le choix du votant est ensuite envoyé au back-end pour traitement.
- **Enregistrement du vote sur la blockchain** : Le back-end prend en charge le traitement du choix du votant et le transmet à la blockchain à travers la fonction `saveChoice()`. Cette étape garantit que le vote est enregistré de manière immuable sur la blockchain, assurant ainsi la transparence et l'intégrité du vote.
- **Confirmation ou échec du vote** : Après que le vote a été enregistré avec succès sur la blockchain, le back-end renvoie une confirmation à l'interface de vote, déclenchant l'affichage de l'écran de succès (`successScreen()`) pour informer le votant que son vote a bien été pris en compte. Si une erreur survient au moment de l'enregistrement du vote, l'interface afficherait un écran d'erreur pour notifier l'échec.

2.3 Outils de développement

La réalisation du système de vote électronique repose sur une combinaison de technologies et d'outils de développement. Cette section présente les principaux outils et technologies utilisés dans le projet.

2.3.1 Blockchain publique Ethereum

Ethereum est une blockchain publique et décentralisée qui permet la création et l'exécution de contrats intelligents, ainsi que le développement d'applications décentralisées (DApps). Contrairement à Bitcoin, qui se concentre principalement sur les transactions de monnaie numérique, Ethereum offre une plateforme flexible pour l'exécution de programmes auto-exécutants appelés smart contracts. Dans notre système de vote, Ethereum est utilisé pour stocker et gérer les votes de manière sécurisée et transparente. Chaque transaction de vote est enregistrée sur la blockchain, assurant ainsi la sécurité et l'intégrité des résultats électoraux grâce à la décentralisation.

2.3.2 Smart Contracts et langage Solidity

Les smart contracts sont des programmes autonomes qui s'exécutent sur la blockchain et automatisent les processus en suivant des règles définies. Ils permettent de gérer des transactions et des processus de manière transparente et sans intermédiaire. Solidity est le langage de programmation principalement utilisé pour écrire ces smart contracts sur Ethereum. Conçu pour être intuitif et puissant, Solidity permet de coder les règles et logiques nécessaires au fonctionnement du système de

vote. Les smart contracts développés en Solidity gèrent les aspects critiques de l'élection, tels que l'enregistrement des votes, la validation des électeurs, et le calcul des résultats.

2.3.3 NodeJS

NodeJS est un environnement d'exécution JavaScript côté serveur, connu pour sa capacité à créer des applications réseau évolutives et performantes. Il repose sur le moteur V8 de Google Chrome et est conçu pour gérer des opérations d'entrée/sortie non-bloquantes, ce qui le rend particulièrement adapté pour des applications nécessitant une haute performance. Dans notre système de vote, Node.js est utilisé pour développer le backend de l'application. Il facilite la gestion des interactions entre les utilisateurs et la blockchain, ainsi que la communication avec les smart contracts via des APIs.

2.3.4 Truffle framework et Ganache

Truffle est un framework de développement Ethereum qui simplifie l'écriture, le test, et le déploiement des smart contracts. Il offre des outils pour gérer les migrations de contrats, exécuter des tests automatisés, et organiser les déploiements sur la blockchain. Ganache, quant à lui, est un simulateur de blockchain Ethereum local fourni par la suite Truffle. Il permet de créer un réseau Ethereum privé pour tester et déboguer les smart contracts sans frais ni risques sur le réseau principal. Ensemble, Truffle et Ganache facilitent le développement et la validation des smart contracts avant leur déploiement réel.

2.3.5 ExpressJS

ExpressJS est un Framework web minimaliste pour Node.js qui simplifie le développement d'applications web et d'APIs. Il offre des fonctionnalités robustes pour le routage des requêtes, la gestion des réponses HTTP, et l'organisation des middlewares. Dans le cadre de notre système de vote, Express.js est utilisé pour construire le serveur backend, gérer les requêtes des utilisateurs, et fournir des interfaces API permettant l'interaction avec les smart contracts. Sa flexibilité et sa simplicité en font un outil idéal pour le développement de l'infrastructure backend de l'application.

2.3.6 Web3.js

Web3.js est une bibliothèque JavaScript qui sert de passerelle entre les applications web traditionnelles et les blockchains Ethereum. Elle permet aux développeurs d'interagir avec un nœud Ethereum local ou distant via HTTP, WebSocket ou IPC, offrant ainsi une interface complète pour communiquer avec la blockchain. Cette bibliothèque propose trois fonctionnalités principales :

- L'interaction avec les smart contracts (déploiement, appel de fonctions, lecture de données)
- L'accès aux informations de la blockchain (blocks, transactions, soldes)
- La gestion des comptes et des signatures numériques

Son API intuitive facilite des opérations complexes comme la création de transactions, la signature de messages ou l'écoute d'événements émis par les smart contracts. Elle est devenue un standard de facto dans le développement d'applications décentralisées (dApps) sur Ethereum.

2.3.7 Remix IDE

Remix IDE est un environnement de développement intégré en ligne spécialement conçu pour le développement de smart contracts sur Ethereum. Il offre une interface conviviale pour écrire, tester et déployer des contrats intelligents en Solidity. Remix IDE permet aux développeurs de coder directement dans le navigateur, de compiler leurs contrats, d'exécuter des tests et de déployer des contrats sur des réseaux de test Ethereum ou le réseau principal. Grâce à ses fonctionnalités intégrées telles que le débogage, l'analyse de code et les simulations de transactions, Remix IDE facilite le développement et la vérification des smart contracts avant leur déploiement définitif.

2.3.8 MetaMask

MetaMask est un portefeuille numérique permettant aux utilisateurs d'interagir avec des blockchains compatibles avec les contrats intelligents, notamment Ethereum. Dans le cadre de ce projet, MetaMask est utilisé pour permettre aux autorités électorales de s'authentifier via leurs adresses blockchain. Cette solution leur offre un accès sécurisé aux fonctionnalités des smart contracts présents, leur permettant ainsi d'exploiter les avantages des technologies décentralisées tout en garantissant l'intégrité et la sécurité des processus électoraux.

2.3.9 Visual Studio Code

Visual Studio Code (VSCode) est un éditeur de code source puissant et léger, largement utilisé dans le développement logiciel. Il offre une grande flexibilité grâce à ses extensions et ses fonctionnalités de personnalisation. Pour le développement de smart contracts Ethereum, VSCode est souvent utilisé en combinaison avec des extensions telles que Solidity pour la syntaxe et la compilation des contrats intelligents, ainsi que des outils comme Truffle et Ganache pour le déploiement et les tests. VSCode permet aux développeurs de gérer efficacement le code source, d'intégrer des systèmes de contrôle de version, et d'automatiser les tâches de développement avec des outils de build et de débogage.

Conclusion

Ce chapitre a détaillé l'architecture du système de vote électronique, en exposant les rôles des différents acteurs et les composants essentiels. Nous avons exploré les outils et technologies utilisés, tels que la blockchain Ethereum, les smart contracts en Solidity, et les Frameworks de développement comme Truffle et Ganache.
En conclusion, ces éléments fournissent une base solide pour la mise en œuvre du système. Le prochain chapitre abordera la modélisation et le développement du système de vote, avec une explication approfondie des étapes de conception.

Chapitre 3

Résultats et discussion

Introduction

Ce chapitre se concentre sur la présentation des résultats du développement du système de vote basé sur la blockchain. Nous commencerons par décrire la conception du smart contract, en mettant en évidence ses fonctionnalités clés pour garantir la sécurité du processus électoral. Nous poursuivrons avec une analyse des résultats obtenus et une discussion sur leurs implications, ainsi que sur les limites du système.

3.1 Fonctionnement général du système

Le système de vote électronique basé sur la blockchain permet de garantir un processus électoral sécurisé et transparent. Cette section décrit les principaux composants et explique comment ils interagissent pour assurer l'efficacité du vote.

3.1.1 Présentation du Smart Contract

Le contrat **EVoting**, présenté dans cette section, est conçu pour gérer les élections, les candidats et les votes. Nous allons explorer les variables et structures définies dans ce contrat, ainsi que leur rôle dans la gestion des élections électroniques.

Variables et Structures :

Le contrat EVoting utilise plusieurs structures et variables pour gérer les élections et les votants :

- **Variables d'état :**
  - `address[] public electoralAuthorities` : Ce tableau contient les adresses des autorités électorales qui sont les seules à pouvoir créer et gérer des élections.
  - `uint public electionCount` : Cette variable garde une trace du nombre total d'élections créées sur la plateforme.
  - `enum ElectionStatus` : Un énuméré pour définir trois statuts possibles pour une élection : `PENDING`, `OPEN` et `CLOSED`.

  - `mapping(uint => ElectionDetails) public elections` : Un mapping qui associe chaque élection (identifiée par un entier `uint`) à ses détails (`ElectionDetails`).

- **Structures de données :**
  - `struct Candidate` : Représente un candidat dans une élection. Elle contient :
    * `uint id` : L'identifiant unique du candidat.
    * `string name` : Le nom du candidat.
    * `string program` : Le programme du candidat.
    * `uint voteCount` : Le nombre de votes reçus par le candidat.
  - `struct Voter` : Représente un votant potentiel, avec les informations suivantes :
    * `uint id` : L'identifiant unique du votant.
    * `string name` : Le nom du votant.
    * `string surname` : Le prénom du votant.
    * `string email` : L'adresse email du votant.
    * `bool hasVoted` : Indique si le votant a déjà voté.
  - `struct ElectionDetails` : Représente les détails d'une élection, et comprend :
    * `uint id` : L'identifiant unique de l'élection.
    * `string name` : Le nom de l'élection.
    * `string description` : Une description de l'élection.
    * `uint startDate` : La date de début de l'élection.
    * `uint endDate` : La date de fin de l'élection.
    * `bool isPrivate` : Un booléen qui indique si l'élection est privée ou publique.
    * `bool votingSessionOpen` : Un booléen qui indique si la session de vote est ouverte.
    * `ElectionStatus status` : Le statut actuel de l'élection.
    * `Candidate[] candidates` : Un tableau de candidats pour cette élection.
    * `Voter[] voters` : Un tableau des votants autorisés (pour les élections privées).

Modifiers :

- `onlyElectoralAuthority` : Restreint l'accès aux fonctions aux seules autorités électorales. Vérifie que l'adresse de l'appelant est dans `electoralAuthorities`.
- `electionExists` : Vérifie que l'élection existe en s'assurant que l'ID de l'élection est valide (supérieur à zéro et inférieur ou égal à `electionCount`).

Fonctions principales :

- **createElection(string memory _name, string memory _description, uint _startDate, uint _endDate, bool _isPrivate, Voter[] memory _voters)**
  Cette fonction permet aux autorités électorales de créer une nouvelle élection. Elle prend en paramètres le nom, la description, les dates de début et de fin, un booléen indiquant si l'élection est privée, ainsi qu'une liste de votants pour les élections privées. Un événement `ElectionCreated` est émis après la création de l'élection. Pour plus de détails sur son implémentation, veuillez consulter l'Annexe 3.2.

- **addCandidate(uint _electionId, string memory _name, string memory _program)**
  Cette fonction permet d'ajouter un candidat à une élection existante. Elle prend en paramètres l'ID de l'élection, le nom du candidat et son programme, puis l'ajoute à la liste des candidats de l'élection. Un événement `CandidateAdded` est émis après l'ajout du candidat. Pour plus de détails sur son implémentation, veuillez consulter l'Annexe 3.2.

- **vote(uint _electionId, uint _candidateId, string memory _name, string memory _surname, string memory _email)**
  Cette fonction permet à un utilisateur de voter pour un candidat. Elle vérifie que le votant est autorisé (dans le cas des élections privées) et que le candidat sélectionné est valide. Si toutes les conditions sont respectées, le vote est enregistré et un événement `Voted` est émis. Pour plus de détails sur son implémentation, veuillez consulter l'Annexe 3.2.

- **openVotingSession(uint _electionId)**
  Cette fonction ouvre la session de vote d'une élection, en changeant son statut à `OPEN`. Elle est restreinte aux autorités électorales. Pour plus de détails sur son implémentation, veuillez consulter l'Annexe 3.2.

- **closeVotingSession(uint _electionId)**
  Cette fonction ferme la session de vote d'une élection, en changeant son statut à `CLOSED`. Elle est également réservée aux autorités électorales. Pour plus de détails sur son implémentation, veuillez consulter l'Annexe 3.2.

- **getElectionDetails(uint _electionId)**
  Cette fonction retourne les détails d'une élection, y compris son nom, sa description, ses dates, le statut privé/public, les candidats et les votants. Pour plus de détails sur son implémentation, veuillez consulter l'Annexe 3.2.

- **getOnGoingElections()**
  Cette fonction renvoie une liste des élections en cours, c'est-à-dire celles dont la session de vote est actuellement ouverte. Pour plus de détails sur son implémentation, veuillez consulter l'Annexe 3.2.

- **getCompletedElections()**
  Cette fonction renvoie une liste des élections qui sont terminées, avec un statut `CLOSED`. Pour plus de détails sur son implémentation, veuillez consulter l'Annexe 3.2.

Ce contrat utilise le langage Solidity pour garantir la sécurité des transactions et des données. Les structures et fonctions définies permettent de gérer efficacement les processus électoraux tout en offrant la flexibilité nécessaire pour différents types d'élections.

Tests unitaires et déploiement sur Ganache

Les tests unitaires sont cruciaux pour garantir le bon fonctionnement du contrat intelligent avant son déploiement sur la blockchain. Cette section présente le processus de création et d'exécution des tests unitaires, ainsi que le déploiement du contrat sur Ganache, une blockchain locale pour le développement Ethereum.

Déploiement sur Ganache

Avant d'exécuter les tests unitaires, il est nécessaire de déployer le contrat sur un environnement de développement local. Ganache a été utilisé pour simuler un réseau Ethereum.
Le processus débute avec la commande de déploiement suivante, qui migre le contrat EVoting.sol sur la blockchain locale de Ganache :

```
truffle migrate --network development
```

La figure 3.1 illustre le résultat de cette migration, indiquant l'adresse du contrat déployé ainsi que le résumé des transactions associées.

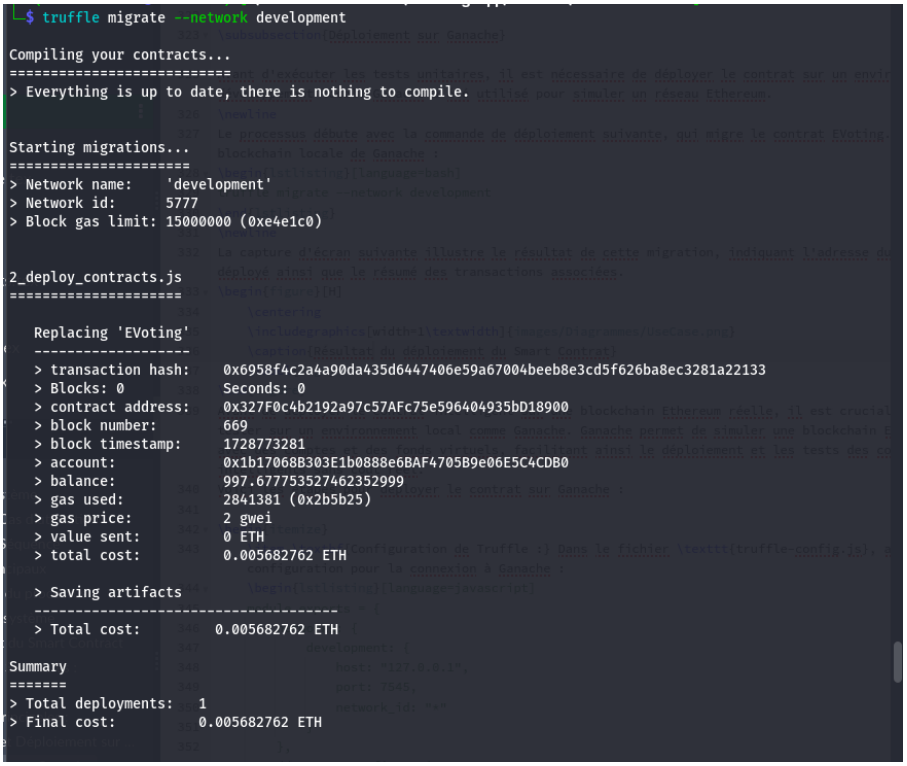


FIGURE 3.1 : Résultat du déploiement du Smart Contrat

Une fois le contrat déployé, les tests unitaires peuvent être exécutés pour vérifier que les différentes fonctionnalités du contrat fonctionnent correctement. Ces tests sont effectués à l'aide des scripts suivants, intégrés dans le framework Truffle.

Description des tests effectués

Les tests effectués sur le smart contrat couvrent plusieurs aspects essentiels du système électoral. Ils vérifient d'abord la gestion des autorités électorales, en s'assurant que seul l'administrateur peut ajouter de nouvelles autorités et que les utilisateurs non autorisés ne peuvent pas le faire.
Ensuite, des tests de création d'élections garantissent que des élections publiques et privées peuvent être créées correctement avec les détails appropriés. La gestion des candidats est également testée, confirmant que seuls les autorités électorales peuvent ajouter des candidats. Le processus de vote est validé, s'assurant que les votants ne peuvent pas voter lorsque la session est fermée et que les votes sont correctement enregistrés pour les candidats valides.
Enfin, des tests de gestion des élections vérifient que les sessions de vote peuvent être ouvertes et fermées par le propriétaire, et permettent de récupérer les élections en cours et celles qui sont terminées.

Résultats des tests unitaires

Une fois les tests lancés avec la commande :

```
truffle test
```

Les résultats des tests sont obtenus, comme le montre la figure 3.2, démontrant que toutes les fonctionnalités ont été correctement implémentées et validées.

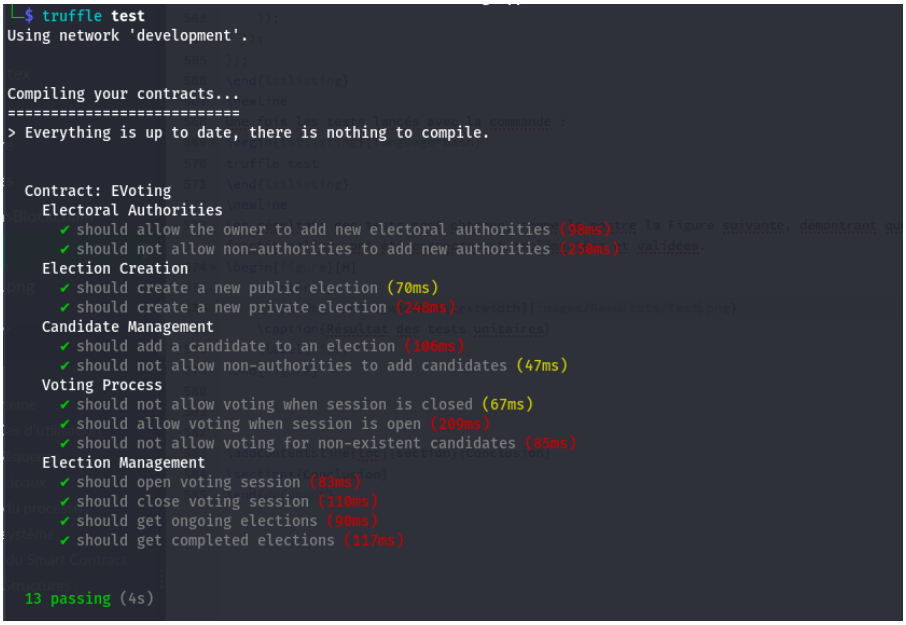


FIGURE 3.2 : Résultat des tests unitaires

Ce processus de déploiement et de test illustre la solidité du contrat et la bonne intégration de ses fonctionnalités dans un environnement de développement avant de passer à une phase de déploiement sur un réseau Ethereum réel.

3.1.2 Scénarios d'utilisation

Dans cette section, nous illustrons le fonctionnement de l'application à travers plusieurs scénarios d'utilisation représentatifs. Ces scénarios mettent en évidence les principales fonctionnalités de l'application, à savoir : l'authentification des utilisateurs, la création d'élections, le processus de vote et l'affichage des résultats.
Chaque scénario sera accompagné de captures d'écran pour démontrer l'interface utilisateur ainsi que les différentes interactions possibles au sein du système. Ces démonstrations visuelles permettent de comprendre concrètement le déroulement de chaque action dans l'application, tout en mettant en lumière les avantages et la simplicité d'utilisation de ce système de vote électronique basé sur la blockchain.

Scénario d'authentification

L'authentification des utilisateurs est un processus clé pour garantir la sécurité et la personnalisation des actions au sein de l'application. Ce scénario présente les trois étapes principales du processus : l'inscription (signup), la connexion via un email et un mot de passe (login simple), et la connexion via MetaMask (login MetaMask).

- **Inscription (Signup) :**
  - **Description :** L'utilisateur remplit un formulaire d'inscription contenant les champs requis. Après la soumission, les données sont envoyées au backend qui vérifie si l'email et l'adresse Ethereum sont déjà associés à un compte existant. Si tout est correct, un nouveau compte est créé.
  - **Fonction associée :** La route `POST /signup` gère la création de nouveaux utilisateurs, comme décrit dans l'annexe dédiée aux routes API.
  - **Réponse attendue :** Si l'inscription est réussie, le backend renvoie un jeton JWT et les informations de l'utilisateur. En cas d'échec (email ou adresse Ethereum déjà utilisés), un message d'erreur est retourné.

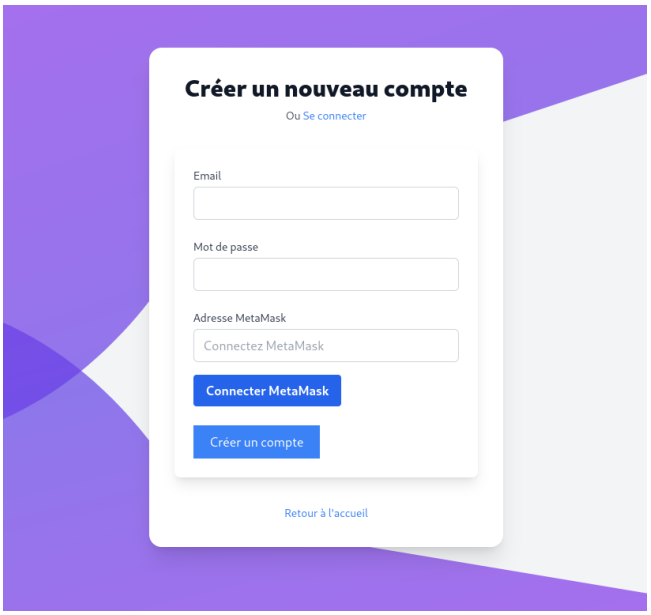


FIGURE 3.3 : Formulaire d'inscription avec email, mot de passe et adresse Ethereum.

- **Connexion simple (Login) :**
  - **Description :** L'utilisateur soumet son email et son mot de passe. Le backend vérifie si les informations sont correctes et génère un jeton JWT pour autoriser l'accès.
  - **Fonction associée :** La route `POST /login`, déjà expliquée dans l'annexe dédiée aux API, est responsable de l'authentification via email et mot de passe.
  - **Réponse attendue :** Si la connexion est réussie, le backend renvoie un jeton JWT et les informations de l'utilisateur. En cas d'échec (mauvais mot de passe ou email inexistant), un message d'erreur est retourné.

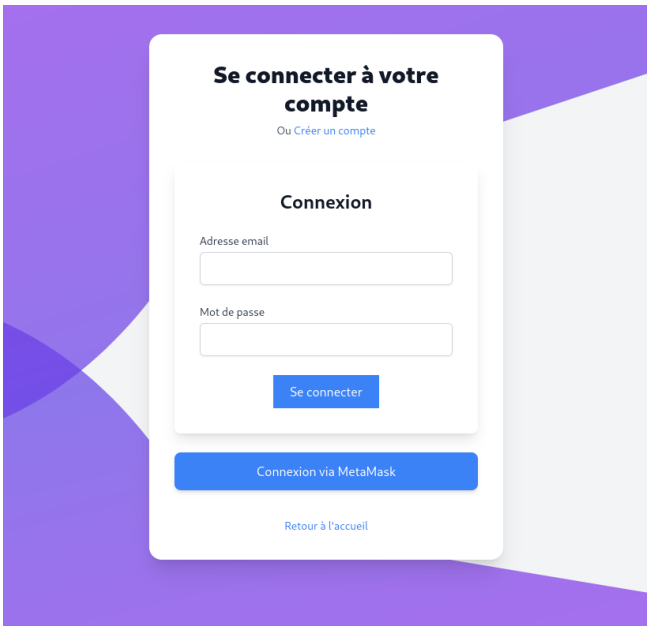


FIGURE 3.4 : Formulaire de connexion classique par email et mot de passe.

- **Connexion via MetaMask (Login MetaMask) :**
  - **Description :** L'utilisateur initie le processus de connexion en fournissant son adresse Ethereum. Le backend génère un nonce unique qui doit être signé via MetaMask pour prouver la propriété de l'adresse. Le **nonce** est une valeur unique générée pour chaque session de connexion afin de garantir l'authenticité et la sécurité de l'utilisateur lors de l'interaction avec le portefeuille blockchain (**MetaMask**). Une fois la signature vérifiée, l'utilisateur est authentifié.
  - **Fonction associée :** Les routes `POST /request-nonce` et `POST /login-metamask` sont utilisées pour demander un nonce et valider la signature du message respectivement, comme décrit dans l'annexe dédiée aux API.
  - **Réponse attendue :** Si la signature est correcte, le backend renvoie un jeton JWT et les informations de l'utilisateur. Si la signature est invalide ou si l'utilisateur n'est pas enregistré, un message d'erreur est retourné.

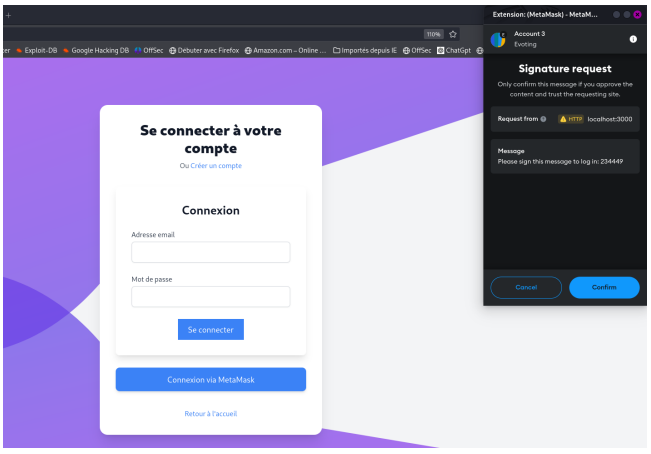


FIGURE 3.5 : Fenêtre MetaMask pour signer le message avec le nonce lors de la connexion.

Scénario de création d'élection

Ce scénario illustre le processus complet de création d'une élection à travers l'interface utilisateur de l'application. Une autorité électorale, après authentification, peut créer une élection en définissant plusieurs paramètres tels que le titre, la description, les dates d'ouverture et de fermeture du scrutin, ainsi que les candidats.

- **Description** : L'autorité électorale commence par remplir un formulaire de création d'élection. Ce formulaire permet de renseigner toutes les informations nécessaires concernant l'élection : le titre de l'élection, sa description, les dates de début et de fin, ainsi que les candidats qui concourent.

  Dans le cadre de certaines élections, notamment privées, une étape supplémentaire est nécessaire. L'autorité doit inclure un fichier CSV contenant la liste des votants autorisés à participer, ce qui permet de garantir que seules les personnes autorisées pourront voter.

  Une fois tous les champs renseignés, le formulaire est soumis, et les informations sont envoyées au backend, qui se charge de la gestion de ces données et de l'enregistrement des informations de l'élection sur la blockchain.

- **Fonction associée** : La route `POST /elections`, décrite dans l'annexe dédiée aux routes API, est utilisée pour la création d'une élection. Le backend valide les informations fournies et enregistre les détails de l'élection, ainsi que la liste des votants dans le cas d'une élection privée. Si un fichier CSV est inclus, il est analysé pour extraire les adresses Ethereum des votants.

- **Réponse attendue** : En cas de succès, la réponse inclut une confirmation de la création de l'élection, avec les détails de celle-ci (titre, description, dates, candidats). Si l'élection est privée, la liste des votants issus du fichier CSV est également confirmée. Si des erreurs surviennent (par exemple, un fichier CSV mal formaté ou des dates invalides), un message d'erreur approprié est renvoyé à l'utilisateur.

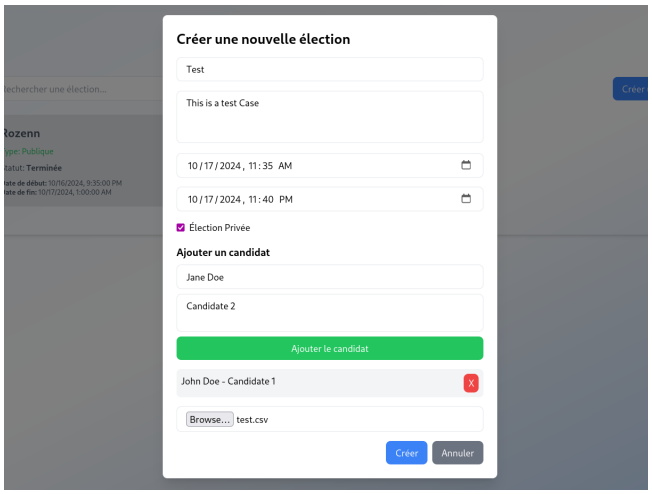


FIGURE 3.6 : Formulaire de création d'une élection avec champ pour le fichier CSV (si applicable).

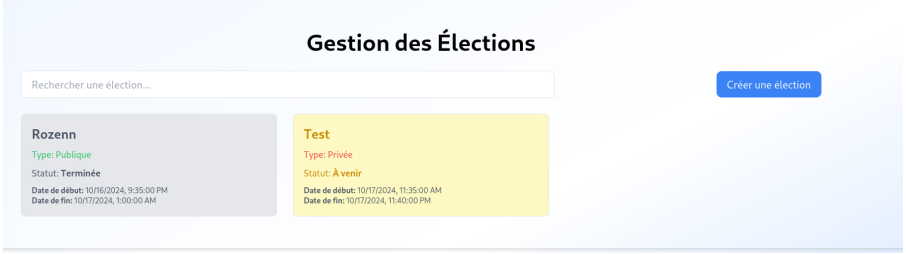


FIGURE 3.7 : Confirmation de la création d'une élection.

Scénario de vote en élection privée

Ce scénario décrit le processus de vote lors d'une élection privée, où les votants doivent être préalablement enregistrés et autorisés à participer.

- **Description** : Lorsque l'élection est ouverte, les votants autorisés se connectent à l'application. Ils accèdent à la page dédiée au vote, où ils peuvent choisir parmi les élections disponibles auxquelles ils sont habilités à participer.

  Pour voter, le votant sélectionne l'élection privée souhaitée et remplit un formulaire contenant ses informations préenregistrées, telles que son nom, son adresse e-mail et son identifiant (ID). Dans le contexte béninois, l'ID pourrait être remplacé par le **Numéro Personnel d'Identification (NPI)**, qui est utilisé pour identifier les citoyens.

  Après avoir rempli le formulaire avec ces informations, le votant choisit son candidat préféré en cochant la case correspondante. Avant de finaliser son vote, un récapitulatif de ses choix est affiché pour confirmation.

  Une fois la sélection validée, le vote est soumis au backend. Le système envoie la transaction à la blockchain, assurant ainsi l'enregistrement sécurisé du vote. En guise de preuve, une notification est générée pour informer le votant que son vote a été soumis avec succès.

- **Fonction associée** : Le processus de vote utilise la route `POST /vote`, décrite dans l'annexe dédiée aux routes API, qui traite la soumission du vote et interagit avec le smart contract pour enregistrer le choix du votant. Le système effectue une vérification pour s'assurer que l'utilisateur est autorisé à voter dans l'élection sélectionnée.

- **Réponse attendue** : Après la soumission du vote, le système renvoie une réponse confirmant que le vote a été enregistré avec succès. Si une erreur se produit (comme une tentative de vote par un utilisateur non autorisé), un message d'erreur approprié est fourni.

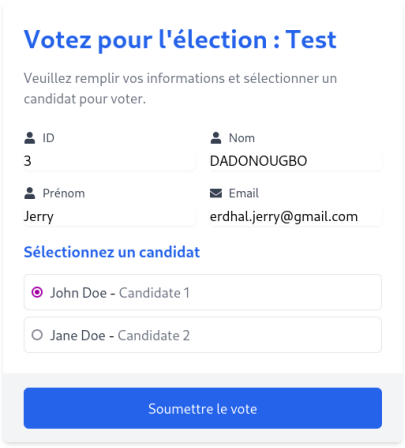


FIGURE 3.8 : Page de vote pour une élection privée

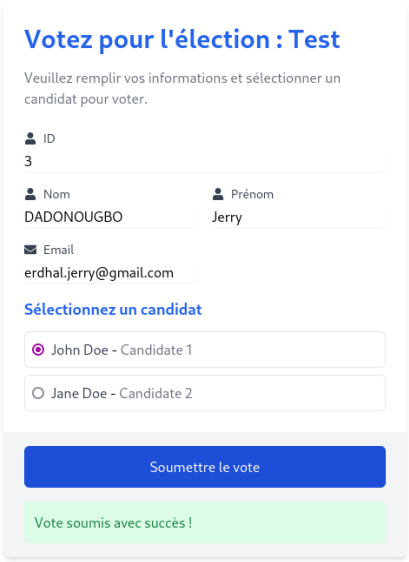


FIGURE 3.9 : Confirmation du vote soumis pour l'élection privée.

Scénario d'affichage des résultats

Ce scénario décrit le processus d'affichage des résultats d'une élection après la clôture des votes.

- **Description** : Une fois l'élection terminée, les résultats doivent être publiés pour que tous les participants et observateurs puissent les consulter. L'utilisateur peut accéder à la section des résultats directement à partir de l'interface de l'application.

  Dans cette section, l'utilisateur peut sélectionner l'élection dont il souhaite voir les résultats. L'application interroge le backend pour récupérer les données des résultats stockées sur la blockchain. Les résultats affichés présentent un tableau récapitulatif des candidats et le nombre de votes qu'ils ont reçus.

  Le tableau présente chaque candidat avec les informations suivantes :

  - Nom du candidat
  - Nombre de votes reçus

- **Fonction associée** : Le processus d'affichage des résultats utilise la route `GET /results/:electionId`, comme décrite dans l'annexe dédiée aux routes API, qui permet de récupérer les résultats de l'élection spécifiée en interrogeant le smart contract. Cette fonction assure que seuls les utilisateurs autorisés peuvent consulter les résultats, en validant leur identité au préalable.
- **Réponse attendue** : Après la demande des résultats, le système renvoie une réponse contenant les données de l'élection, y compris le nombre total de votes et le tableau récapitulatif des votes par candidat. Si aucune élection n'est trouvée ou si l'utilisateur n'est pas autorisé à accéder aux résultats, un message d'erreur approprié est fourni.

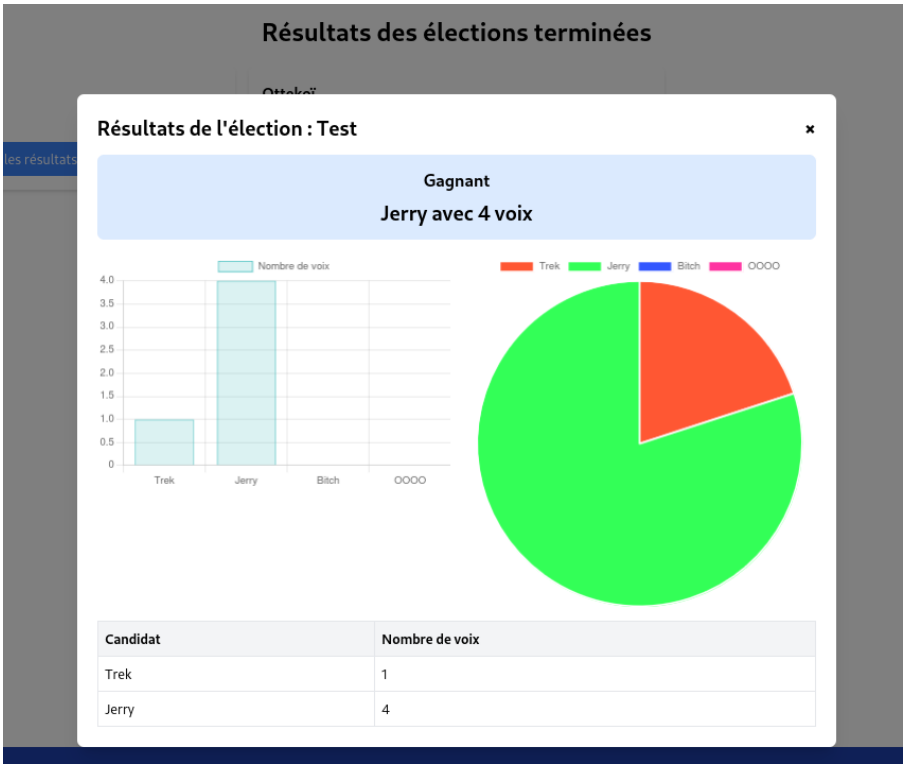


FIGURE 3.10 : Tableau récapitulatif des votes des candidats.

3.2 Discussion

Le système de vote électronique basé sur la blockchain développé dans ce projet présente des avantages indéniables, notamment en matière de transparence, de sécurité et d'immuabilité des données. Cependant, il est crucial de reconnaître certaines limites qui pourraient entraver son adoption généralisée.

L'un des principaux défis réside dans **les coûts de déploiement** associés à l'utilisation de la blockchain. Les frais de transaction sur des réseaux comme Ethereum peuvent fluctuer considérablement, surtout lors des périodes de forte activité. Ces coûts peuvent devenir prohibitifs, en particulier pour des organisations ou des institutions publiques qui doivent gérer des budgets limités. Par conséquent, une évaluation minutieuse des coûts doit être effectuée pour garantir que l'implémentation du système soit économiquement viable.

Une autre limitation majeure concerne la **nécessité d'une connexion Internet stable** pour participer

au vote. Cette dépendance technologique peut créer une fracture numérique, excluant potentiellement les citoyens vivant dans des zones mal desservies ou ceux ne disposant pas d'un accès Internet fiable. Pour remédier à cette situation, il serait judicieux d'envisager la mise en place de bureaux de vote équipés d'une connexion Internet sécurisée et de dispositifs dédiés, permettant ainsi à tous les électeurs d'exercer leur droit de vote, indépendamment de leur situation personnelle.
L'accessibilité du système constitue également un enjeu crucial, particulièrement pour les personnes en situation de handicap ou celles confrontées à des difficultés de lecture. Pour garantir un accès universel au vote, le système devrait intégrer des **fonctionnalités d'assistance adaptées**, telles que :

- Une interface vocale pour les personnes malvoyantes ou non-voyantes
- Des options de grossissement de texte et de contraste élevé
- Une assistance humaine formée disponible dans les bureaux de vote
- Des interfaces simplifiées et intuitives pour les personnes ayant des difficultés de lecture
- Des supports multilingues pour les communautés linguistiques diverses

La mise en production d'un système de vote électronique nécessite une préparation rigoureuse. Il est essentiel de s'assurer que l'infrastructure technique soit suffisamment robuste pour gérer un volume élevé de transactions durant les élections. De plus, des tests approfondis doivent être réalisés pour identifier et corriger les problèmes potentiels avant le déploiement. La formation des utilisateurs, des administrateurs et du personnel d'assistance est également indispensable pour assurer une utilisation efficace et sécurisée du système.
Enfin, il est important de noter que l'introduction d'un tel système nécessite une sensibilisation et une éducation du public pour qu'il puisse comprendre et faire confiance à cette nouvelle technologie. Les préoccupations liées à la sécurité et à la confidentialité des données doivent être adressées de manière proactive pour renforcer la confiance des électeurs. Cette approche inclusive, combinant innovation technologique et accessibilité universelle, permettrait de moderniser le processus électoral tout en garantissant la participation démocratique la plus large possible.

Conclusion

Ce chapitre a présenté une analyse des résultats obtenus avec l'application de vote électronique basée sur la blockchain, illustrant les scénarios d'utilisation de l'authentification, de la création d'élections, du vote et de l'affichage des résultats. Nous avons démontré l'efficacité du processus de vote pour les électeurs et les administrateurs, tout en fournissant des captures d'écran pour illustrer la convivialité de l'application.
Nous avons également mis en lumière les principales fonctionnalités de l'application, notamment les mécanismes d'authentification, le développement du backend et des API, ainsi que le smart contract qui assure la transparence et la fiabilité des résultats. L'intégration de fonctionnalités d'accessibilité démontre l'engagement du projet envers une participation inclusive.
Cependant, certains défis, tels que les coûts de déploiement, la nécessité d'une connexion Internet stable et le besoin d'une sensibilisation accrue, demeurent cruciaux à considérer. En somme, bien que cette technologie offre des opportunités prometteuses pour moderniser le processus électoral, une

mise en œuvre réfléchie intégrant des solutions d'accessibilité universelle reste essentielle pour son succès.

Conclusion Générale et Perspectives

Ce mémoire a présenté le développement d'une application de vote électronique basée sur la blockchain, visant à rendre les élections plus transparentes et fiables. Nous avons exploré la conception de l'application, des interfaces utilisateur, des composants backend et des smart contracts, en mettant l'accent sur la sécurité et l'accessibilité du système.

À travers différents scénarios d'utilisation, nous avons montré comment l'application permet aux utilisateurs de s'authentifier, de créer des élections, de voter et d'afficher les résultats facilement. L'intégration de fonctionnalités d'assistance et d'accessibilité démontre notre engagement à rendre le vote électronique accessible à tous les citoyens, indépendamment de leurs capacités techniques ou de leurs situations personnelles.

Cependant, il est important de noter que des défis subsistent, notamment en ce qui concerne les coûts de déploiement, la nécessité d'une connexion Internet stable et l'importance d'une infrastructure adaptée. Ces aspects doivent être soigneusement évalués pour assurer la viabilité et l'inclusivité du système.

En somme, ce projet ouvre la voie à des élections modernes, sécurisées et accessibles à tous. Nous espérons que cette approche, combinant innovation technologique et accessibilité universelle, inspirera d'autres initiatives visant à améliorer les processus électoraux tout en garantissant une participation démocratique élargie.

Annexe A : Code source du Smart Contract

Fonction createElection

```
        function createElection(
        string memory _name,
        string memory _description,
        uint _startDate,
        uint _endDate,
        bool _isPrivate,
        Voter[] memory _voters  // Ajout de la liste des votants pour les élections privées
    ) public onlyElectoralAuthority returns (uint) {
        require(_endDate > _startDate, "La date de fin doit etre superieure a la date de debut.");
        require(_startDate >= block.timestamp, "La date de debut doit etre dans le futur.");

        electionCount++;

        ElectionDetails storage newElection = elections[electionCount];
        newElection.id = electionCount;
        newElection.name = _name;
        newElection.description = _description;
        newElection.startDate = _startDate;
        newElection.endDate = _endDate;
        newElection.isPrivate = _isPrivate;
        newElection.status = ElectionStatus.PENDING;
        newElection.votingSessionOpen = false;

        if (_isPrivate) {
            for (uint i = 0; i < _voters.length; i++) {
                newElection.voters.push(_voters[i]);
            }
        }

        emit ElectionCreated(electionCount, _name, _isPrivate, _startDate, _endDate);
        return electionCount;
    }
```

Fonction addCandidate

```
    function addCandidate(
        uint _electionId,
        string memory _name,
        string memory _program
    ) public onlyElectoralAuthority electionExists(_electionId) {
        ElectionDetails storage election = elections[_electionId];
        election.candidates.push(Candidate(election.candidates.length + 1, _name, _program, 0));

        emit CandidateAdded(_electionId, election.candidates.length, _name, _program);
    }
```

Fonction Vote

```
    function vote(
        uint _electionId,
        uint _candidateId,
        string memory _name,
        string memory _surname,
```

```
        string memory _email
    ) public electionExists(_electionId) {
        ElectionDetails storage election = elections[_electionId];

        require(election.votingSessionOpen, "La session de vote n'est pas ouverte.");
        require(_candidateId > 0 && _candidateId <= election.candidates.length, "Candidat invalide.");

        if (election.isPrivate) {
            bool isAuthorized = false;
            for (uint i = 0; i < election.voters.length; i++) {
                if (keccak256(abi.encodePacked(election.voters[i].email)) == keccak256(abi.encodePacked(_email)))
                    require(!election.voters[i].hasVoted, "Le votant a deja vote.");
                    election.voters[i].hasVoted = true;
                    isAuthorized = true;
                    break;
                }
            }
            require(isAuthorized, "Non autorise a voter dans cette election privee.");
        } else {
            election.voters.push(Voter(election.voters.length + 1, _name, _surname, _email, true));
        }

        election.candidates[_candidateId - 1].voteCount++;

        emit Voted(_electionId, election.voters.length, _candidateId);
    }
```

Fonction openVotingSession

```
    function openVotingSession(uint _electionId) public onlyElectoralAuthority electionExists(_electionId) {
        ElectionDetails storage election = elections[_electionId];
        require(election.status == ElectionStatus.PENDING, "L'election est deja ouverte ou terminee.");

        election.votingSessionOpen = true;
        election.status = ElectionStatus.OPEN;
    }
```

Fonction closeVotingSession

```
    function closeVotingSession(uint _electionId) public onlyElectoralAuthority electionExists(_electionId) {
        ElectionDetails storage election = elections[_electionId];
        require(election.status == ElectionStatus.OPEN, "L'election n'est pas ouverte.");

        election.votingSessionOpen = false;
        election.status = ElectionStatus.CLOSED; // Changer l'état+FFFD"CLOSED"
    }
```

Fonction getElectionDetails

```
    function getElectionDetails(uint _electionId) public view electionExists(_electionId) returns (
        string memory name,
        string memory description,
        bool isPrivate,
        bool votingSessionOpen,
        uint startDate,
        uint endDate,
        Candidate[] memory candidates,
        Voter[] memory voters
    ) {
```

```
        ElectionDetails storage election = elections[_electionId];
        return (
            election.name,
            election.description,
            election.isPrivate,
            election.votingSessionOpen,
            election.startDate,
            election.endDate,
            election.candidates,
            election.voters
        );
    }
```

Fonction getOnGoingElections

```
    function getOnGoingElections() public view returns (uint[] memory) {
        uint count = 0;
        for (uint i = 1; i <= electionCount; i++) {
            if (elections[i].votingSessionOpen) {
                count++;
            }
        }

        uint[] memory ongoingElections = new uint[](count);
        uint index = 0;
        for (uint i = 1; i <= electionCount; i++) {
            if (elections[i].votingSessionOpen) {
                ongoingElections[index] = i;
                index++;
            }
        }

        return ongoingElections;
    }
```

Fonction getCompletedElections

```
    function getCompletedElections() public view returns (uint[] memory) {
        uint count = 0;

        // Compter le nombre d'élections qui ont le statut CLOSED
        for (uint i = 1; i <= electionCount; i++) {
            if (elections[i].status == ElectionStatus.CLOSED) {
                count++;
            }
        }

        // Créer un tableau pour stocker les IDs des élections terminées
        uint[] memory completedElections = new uint[](count);
        uint index = 0;

        // Remplir le tableau avec les élections fermées
        for (uint i = 1; i <= electionCount; i++) {
            if (elections[i].status == ElectionStatus.CLOSED) {
                completedElections[index] = i;
                index++;
            }
        }

        return completedElections;
    }
```

Annexe B : Présentation des routes APIs pour la gestion des élections

Route : `GET /api/elections`

Cette route récupère la liste des élections dans le système. Elle retourne un tableau contenant des informations telles que le nom, la description, et le statut.

Route : `POST /api/elections/new`

Permet de créer une nouvelle élection en fournissant des détails comme le nom, les dates, et le type d'élection (publique ou privée). Retourne l'ID de l'élection créée.

Route : `GET /api/elections/electionId`

Récupère les détails d'une élection spécifique (nom, description, candidats, etc.) en fonction de son ID.

Route : `POST /api/elections/electionId/candidates`

Ajoute un candidat à une élection existante en fournissant son nom et son programme électoral.

Route : `POST /api/elections/electionId/vote`

Permet à un utilisateur de soumettre son vote pour un candidat donné dans une élection spécifique.

Route : `GET /api/ongoingelections`

Retourne la liste des élections en cours dont la session de vote est ouverte.

Route : `GET /api/completedelections`

Récupère les élections clôturées et leurs résultats.

Route : `GET /api/elections/electionId/results`

Retourne les résultats d'une élection, incluant le nombre de votes pour chaque candidat.

Bibliographie

[1] Commission Électorale Nationale Autonome, *Rapport des élections législatives 2019 au Bénin*, 2023.

[2] Mission d'Observation Électorale de l'Union Africaine, *Rapport des élections législatives au Bénin, 28 avril 2019*, Union Africaine, 2019.

[3] Justicepaix.be, Cycles répétitifs dans l'histoire électorale congolaise : un regard critique, https://www.justicepaix.be/cycles-repetitifs-dans-lhistoire-electorale-congolaise-un-regard-critique/, consulté le 14 Septembre 2024

[4] Cena.sn, Rapport de mission sur les élections générales nigérianes de février 2023, https://www.cena.sn/rapports/RAPPORT-ELECTIONS-NIGERIA.pdf, consulté le 14 Septembre 2024

[5] Ledger.com, Les différentes générations de la blockchain, https://www.ledger.com/fr/academy/les-differentes-generations-de-la-blockchain, consulté le 15 Septembre 2024

[6] Cairn.info, Les systèmes électoraux des républiques au Ghana, Bénin et Cameroun : effets et rapports différenciés, https://shs.cairn.info/article/AFCO_239_0140?lang=fr, consulté le 15 Septembre 2024

[7] Ibm.com, Qu'est-ce que la blockchain ? https://www.ibm.com/fr-fr/topics/blockchain consulté le 16 Septembre 2024

[8] Axiocap.com, 6 exemples d'application de la blockchain, https://www.axiocap.com/blog/6-exemples-application-blockchain, consulté 16 Septembre 2024

[9] Journalducoin.coms, Sierra Leone : élections enregistrées sur une blockchain grâce à Agora, https://journalducoin.com/defi/sierra-leone-elections-enregistrees-blockchain-grace-a-agora/, consulté le 17 Septembre 2024

[10] Wavesenterprise.com, Transparent voting at the Russian Federation of Practical Shooting, https://wavesenterprise.com/use-cases/ipsc-voting, consulté le 17 Septembre 2024

[11] Techstars.com, Voatz : Blockchain Brings the Vote To You, https://www.techstars.com/blog/startup-profile/voatz-blockchain-brings-the-vote-to-you, consulté le 17 Septembre 2024

Table des matières

Dédicace ii

Remerciements iii

Résumé iv

Abstract v

Liste des figures vi

Liste des tableaux vii

Introduction 1

1 Problématique . . . 1
2 Contexte et justification . . . 2
3 Objectifs . . . 2
Organisation du mémoire . . . 3

1 Etat de l'art 4

Introduction . . . 4
1.1 Les systèmes de vote traditionnels . . . 4
1.1.1 Le processus électoral classique en Afrique de l'Ouest . . . 4
1.1.2 Les défis du processus électoral en Afrique de l'Ouest et au Bénin . . . 5
1.2 La blockchain : une technologie émergente . . . 6
1.2.1 Définition de la blockchain et principes de base . . . 6
Première génération : Bitcoin . . . 7
. . . 7
Deuxième génération : Ethereum . . . 7
. . . 7
Troisième génération : Scalabilité et interopérabilité . . . 7
. . . 8
1.2.2 Les applications de la blockchain au-delà des cryptomonnaies . . . 8
1.3 Blockchain et systèmes électoraux . . . 9
1.3.1 Cas d'études : Les initiatives de vote électronique basé sur la blockchain . . . 9
1.3.2 Analyse des résultats et des défis rencontrés dans ces expériences . . . 10
Conclusion . . . 12

2 Méthodes et outils 13

Introduction . . . 13

2.1 Architecture générale du système . . . 13
2.2 Modélisation du système . . . 16
2.2.1 Diagramme de cas d'utilisation . . . 16
. . . 16
2.2.2 Diagramme de séquence . . . 16
. . . 17
2.3 Outils de développement . . . 18
2.3.1 Blockchain publique Ethereum . . . 18
2.3.2 Smart Contracts et langage Solidity . . . 18
2.3.3 NodeJS . . . 19
2.3.4 Truffle framework et Ganache . . . 19
2.3.5 ExpressJS . . . 19
2.3.6 Web3.js . . . 19
2.3.7 Remix IDE . . . 20
2.3.8 MetaMask . . . 20
2.3.9 Visual Studio Code . . . 20
Conclusion . . . 20

3 Résultats et discussion **21**
Introduction . . . 21
3.1 Fonctionnement général du système . . . 21
3.1.1 Présentation du Smart Contract . . . 21
Déploiement sur Ganache . . . 23
. . . 24
Description des tests effectués . . . 24
. . . 24
Résultats des tests unitaires . . . 25
. . . 25
3.1.2 Scénarios d'utilisation . . . 25
3.2 Discussion . . . 31
Conclusion . . . 32

Conclusion **34**

Bibliographie **39**

Table des matières **40**